Proof Complexity for SAT practitioner

BIRS - Theory and Practice of Satisfiability Solving

Massimo Lauria
La Sapienza, Università di Roma

Monday, August 27th 2018

A **starting pack** for proof complexity

- boring for experts
- skip important topics (maybe) not in this program
- you still have to do your own preliminaries

nevertheless **useful**, I hope.

A theory to analyze SAT solvers

Goal: highlight weaknesses of SAT solvers

Methods: mainly lower bounds for proof length

On UNSAT, solving time ⩾ proof length

0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

t=1

0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

t=2

0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

t=3

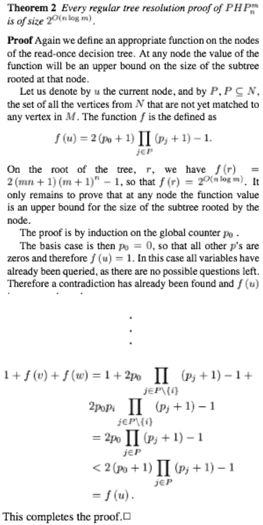
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

t=4

⋮

0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

end



Theorem 2 *Every regular tree resolution proof of PHP_n^m is of size $2^{O(n \log m)}$.*

Proof Again we define an appropriate function on the nodes of the read-once decision tree. At any node the value of the function will be an upper bound on the size of the subtree rooted at that node.

Let us denote by u the current node, and by $P, P \subseteq N$, the set of all the vertices from N that are not yet matched to any vertex in M. The function f is the defined as

$$f(u) = 2(p_0 + 1) \prod_{j \in P} (p_j + 1) - 1.$$

On the root of the tree, r, we have $f(r) = 2(mn + 1)(m + 1)^n - 1$, so that $f(r) = 2^{O(n \log m)}$. It only remains to prove that at any node the function value is an upper bound for the size of the subtree rooted by the node.

The proof is by induction on the global counter p_0.

The basis case is then $p_0 = 0$, so that all other p's are zeros and therefore $f(u) = 1$. In this case all variables have already been queried, as there are no possible questions left. Therefore a contradiction has already been found and $f(u)$

⋮

$$\begin{aligned} 1 + f(v) + f(w) &= 1 + 2p_0 \prod_{j \in P \setminus \{i\}} (p_j + 1) - 1 + \\ & 2p_0 p_i \prod_{j \in P \setminus \{i\}} (p_j + 1) - 1 \\ &= 2p_0 \prod_{j \in P} (p_j + 1) - 1 \\ &< 2(p_0 + 1) \prod_{j \in P} (p_j + 1) - 1 \\ &= f(u). \end{aligned}$$

This completes the proof.□

Proof system for UNSAT

A **polytime** machine $P(\cdot, \cdot)$ so that

- F is in SAT then $P(F, \pi)$ **rejects** for every π
- F is in UNSAT then $P(F, \pi)$ **accepts** for some π

Then π is a refutation of F.

Proof system for UNSAT

A **polytime** machine $P(\cdot, \cdot)$ so that

- F is in SAT then $P(F, \pi)$ **rejects** for every π
- F is in UNSAT then $P(F, \pi)$ **accepts** for some π

Then π is a refutation of F.

Observe: very similar to NP verifier. But proofs of UNSAT can be **much larger** than the formula.

Polysize proofs of UNSAT iff NP $=$ co-NP.

Strength of the proof system

Expressiveness: stronger proofs systems

- stronger SAT solvers
- shorter proofs
- hard to use
- hard to analyze

Simplicity: weaker proofs systems

- weaker SAT solver
- simpler search space
- better heuristics

Outline

1. Proof systems and lower bound techniques
   - resolution
   - cutting planes
   - polynomial calculus
2. Memory and space (quick mention)
3. Extended resolution and DRAT
4. Proof search
   - bounded width/degree proof search
   - non-automatizability

i. resolution proofs

Resolution proof system

Initial CNF

$$F = C_1 \wedge C_2 \wedge \ldots \wedge C_m$$

Rules

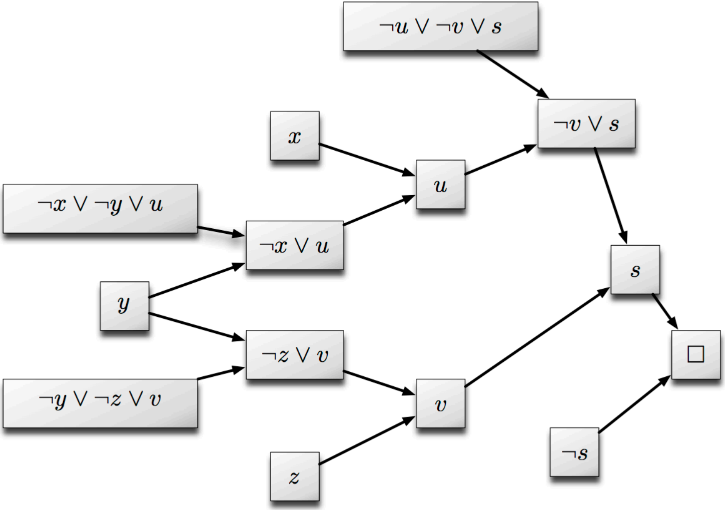
$$\frac{}{C_i} \quad \text{(axiom)} \qquad \frac{A \vee \neg x \qquad x \vee B}{A \vee B} \quad \text{(resolution step)}$$

Refutation

$$\frac{\vdots}{\Box}$$

empty clause ⟶ F is UNSAT

- **Length/Size**: #clauses in the proof (e.g. 14)
- **Width**: largest #literals in clauses (e.g. 3)


$\neg u \vee \neg v \vee s$
$\neg v \vee s$
x
u
$\neg x \vee \neg y \vee u$
$\neg x \vee u$
s
y
$\neg z \vee v$
$\square$
$\neg y \vee \neg z \vee v$
v
$\neg s$
z

CDCL solver and resolution

Theorem (Pipatsrisawat and Darwiche, 2011)

On UNSAT, ***non-deterministic*** *CDCL solver is polynomially equivalent to resolution refutations.*

Width

Width complexity of a derivation $F \vdash D$

$$\min\{w \text{ s.t. } D \text{ has a proof from } F \text{ of width } w\}$$

- large width implies large proof length
- small width implies small size
- possible to study using EF games and expansion

Large width → Large size

Theorem (Ben-Sasson, Wigderson 1999)

Any k-CNF on n variables that requires width w to be refuted also requires refutations of length at least

$$\exp\left(\frac{\Omega(w-k)^2}{n}\right)$$

Large width → Large size

Theorem (Ben-Sasson, Wigderson 1999)

Any k-CNF on n variables that requires width w to be refuted also requires refutations of length at least

$$\exp\left(\frac{\Omega(w-k)^2}{n}\right)$$

Width lower bound $\Omega(n)$ implies size lower bound $2^{\Omega(n)}$.

Small width → small size

Any formula of n variable refutable in width w has a refutation

- of size n^w
- constructible in time $n^{O(w)}$

Proof.

Generates all clauses derivable within width w and check if the empty clause is reached. □

Small size AND large width

There are 3-CNFs on n variables that

- require width $\Omega\left(\sqrt{n}\right)$ to be refuted [BG'99]
- have polynomial size refutation

Small size AND large width

There are 3-CNFs on n variables that

- require width $\Omega\left(\sqrt{n}\right)$ to be refuted [BG'99]
- have polynomial size refutation

Cor 1. cannot improve the exponent of lower bound

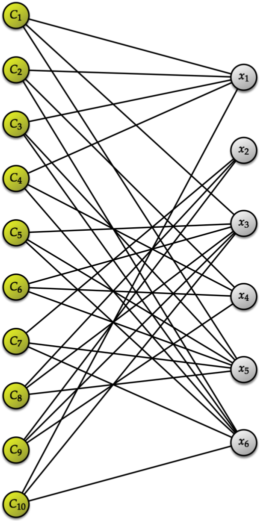
$$\text{proof size } \geqslant \exp\left(\frac{\Omega(w-k)^2}{n}\right)$$

Cor 2. The $n^{O(w)}$ proof search algorithm can be inefficient.

ii. resolution lower bounds 101

Incidence graph of a CNF

$$\begin{array}{ll}
C_1 : & x_1 \vee \overline{x}_3 \vee \overline{x}_5 \\
C_2 : & \overline{x}_1 \vee x_4 \vee \overline{x}_6 \\
C_3 : & x_1 \vee \overline{x}_5 \vee x_6 \\
C_4 : & x_1 \vee \overline{x}_4 \vee \overline{x}_6 \\
C_5 : & x_3 \vee \overline{x}_5 \vee x_6 \\
C_6 : & x_3 \vee x_4 \vee \overline{x}_5 \\
C_7 : & \overline{x}_2 \vee \overline{x}_5 \vee x_6 \\
C_8 : & \overline{x}_2 \vee x_3 \vee \overline{x}_5 \\
C_9 : & \overline{x}_2 \vee \overline{x}_3 \vee x_4 \\
C_{10} & x_1 \vee x_3 \vee \overline{x}_6
\end{array}$$



Neighborhood: $\Gamma(\mathcal{C}) = \bigcup_{C \in \mathcal{C}} \mathrm{Vars}(C)$

CNF formulas with expansion

A CNF formula F is an (r, ϵ)-expander when for every $\mathcal{C} \subseteq F$ with $|\mathcal{C}| \leqslant r$,

$$|\Gamma(\mathcal{C})| \geqslant (1 + \epsilon)|\mathcal{C}| \ .$$

Theorem (Ben-Sasson, Wigderson 1999)

An unsatisfiable k-CNF oven n variable which is an $(\Omega(n), \epsilon)$-expander requires resolution refutations of length $2^{\Omega(n)}$.

Applications of expansion

- Random k-CNFs
- Tseitin formulas
- ”Graph” pigeonhole principle
- …

Feasible interpolation [Krajíček'97]

Proof system P has *feasible interpolation* if, given UNSAT

$$A(\vec{x},\vec{y}) \wedge B(\vec{x},\vec{z})$$

with P-proof π, computes a total function $I(\vec{x})$ so that

$$I(\alpha) = \begin{cases} 0 & \text{only if } A(\alpha,\vec{y}) \text{ is UNSAT} \\ 1 & \text{only if } B(\alpha,\vec{z}) \text{ is UNSAT.} \end{cases}$$

in time/circuit size $|\pi|^{O(1)}$.

Clique vs Coloring formula

Variable sets: graph G, coloring χ, vertex set C

- $\mathrm{Clique}_k(G, C)$: G has a clique C of size k
- $\mathrm{Coloring}_{k-1}(G, \chi)$: G has a coloring χ of size $k - 1$

$\mathrm{Clique}_k(G, C) \wedge \mathrm{Coloring}_{k-1}(G, \chi)$ is unsatisfiable

$$I(G) = \begin{cases} 0 & \text{only if } G \text{ is } k-1\text{-colorable} \\ 1 & \text{only if } G \text{ has the } k\text{-clique.} \end{cases}$$

Clique vs Coloring formula (II)

The interpolant of $\mathrm{Clique}_k(G,C) \wedge \mathrm{Coloring}_{k-1}(G,\chi)$

$$I(G) = \begin{cases} 0 & \text{only if } G \text{ is } k-1\text{-colorable} \\ 1 & \text{only if } G \text{ has the } k\text{-clique.} \end{cases}$$

has monotone circuit size $2^{\sqrt[4]{n}}$. [R'85][AB'97]

Clique vs Coloring formula (II)

The interpolant of $\mathrm{Clique}_k(G, C) \wedge \mathrm{Coloring}_{k-1}(G, \chi)$

$$I(G) = \begin{cases} 0 & \text{only if } G \text{ is } k-1\text{-colorable} \\ 1 & \text{only if } G \text{ has the } k\text{-clique.} \end{cases}$$

has monotone circuit size $2^{\sqrt[4]{n}}$. [R'85][AB'97]

[Krajíček'97] Resolution has **feasible interpolation** and moreover for Clique vs Coloring

- interpolation produces a monotone circuit
- hence, refutation must be large.

iii. cutting planes

Defined by [Chvátal et al ’89]

- based on integer programming [Gomory ’58]
- cardinality constraints

$$\sum_i x_i \leqslant D$$

- PseudoBoolean (E.g. Sat4j, cdcl-cuttingplanes,…)

CNF encoding

A CNF is turned into a system of linear inequalities

$$x \vee y \vee \neg z \qquad \longrightarrow \qquad x + y + (1 - z) \geqslant 1$$

A refutation is a proof of the contradiction

$$0 \geqslant 1$$

Variables: $x_i \in \{0,1\}$

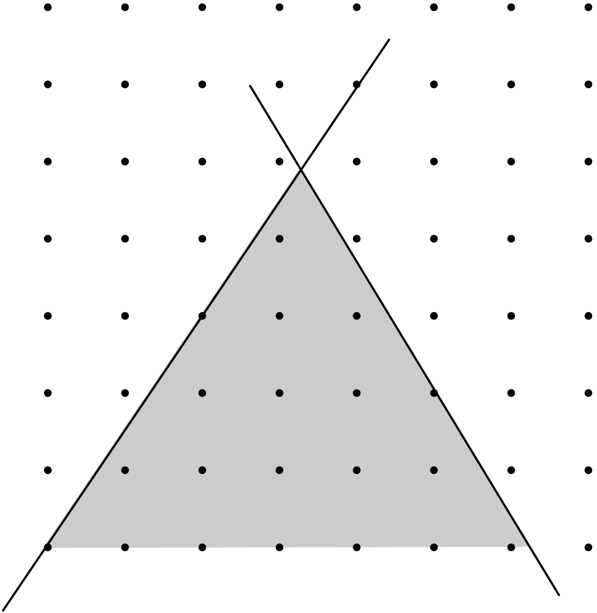
Proof lines: $a_1x_1 + a_2x_2 + \cdots + a_nx_n \leqslant b$ with b and $a_i \in \mathbb{Z}$

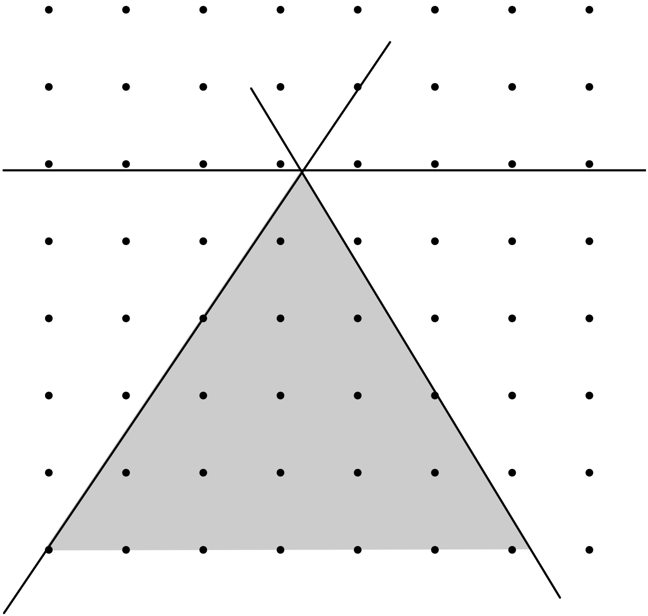
Sum:

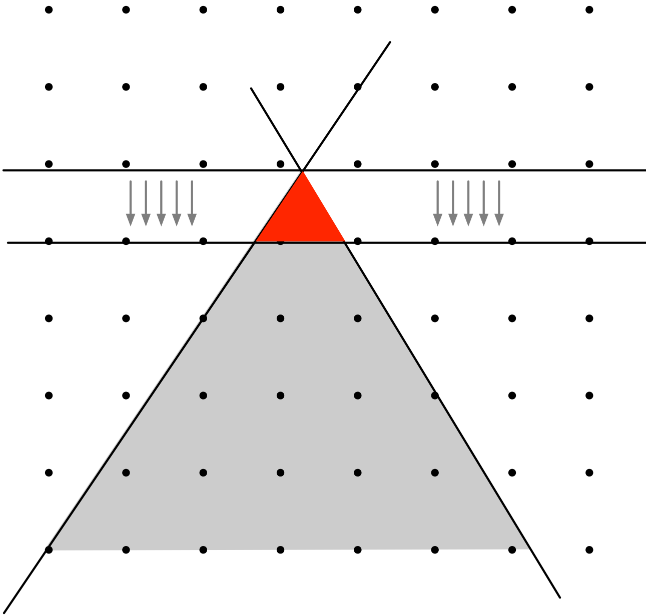
$$\frac{\sum_i a_i x_i \leqslant A \qquad \sum_i a'_i x_i \leqslant A'}{\sum_i (\alpha a_i + \beta a'_i) x_i \leqslant \alpha A + \beta A'} \quad \alpha, \beta \in \mathbb{N}$$

Cut:

$$\frac{\sum_i c a_i x_i \leqslant b}{\sum_i a_i x_i \leqslant \lfloor \frac{b}{c} \rfloor} \quad c \in \mathbb{N}$$







CP lower bound using interpolation

From a CP refutation π of

$$\mathrm{Clique}_k(G, C) \wedge \mathrm{Coloring}_{k-1}(G, \chi)$$

we get **monotone real circuit** of size $\mathrm{poly}(|\pi|)$ for

$$I(G) = \begin{cases} 0 & \text{only if } G \text{ is } k-1\text{-colorable} \\ 1 & \text{only if } G \text{ has the } k\text{-clique.} \end{cases}$$

"$I(G)$ requires large monotone real circuits" [Pudlák'97]

Recent developments

Generalization of interpolation [HP'17][FPPR'17]

Lower bound for $\Theta(\log(n))$-CNF

Lifting + Communication complexity [GGKS'18]

- Assume $F[\vec{x}]$ is a k-CNF of width complexity w
- $x_i \longleftarrow \mathrm{Ind}_m : [m] \times \{0,1\}^m \to \{0,1\}$ with $m = n^\delta$
- $F \circ Ind_m^n$ requires CP refutation of size $n^{\Theta(w)}$.

Is CP a good model for PB solvers?

- if inequalities are encoded as CNF, the solver may behave like resolution
- PB solvers often cannot find short simple CP proofs [Elffers et al.’18]
- maybe weaker proof system are a tighter model [Vinyals et al.’18]

iv. polynomial calculus

Defined by [CEI'96]

- algebraic reasoning
- polynomial equations, ideal membership
- Hilbert's Nullstellensatz
- Gröbner basis computation

CNF encodes as polynomial equations over field $\mathbb{F}$

$$x \vee y \vee \neg z \qquad \longrightarrow \qquad xy(1-z) = 0$$

- 0 encodes **true** and 1 encodes **false**
- boolean axioms $x_i^2 - x_i = 0$ for each variables x_i

Initial CNF

$$F = C_1 \land C_2 \land \ldots \land C_m \longrightarrow f_1 = 0 \; f_2 = 0 \; \ldots \; f_m = 0$$

Rules (preserve common boolean roots)

$$\frac{}{f_j} \qquad \frac{}{x_i^2 - x_i} \qquad \frac{p}{x_i p} \qquad \frac{p \qquad q}{\alpha p + \beta q} \quad \alpha, \beta \in \mathbb{F}$$

Refutation

$$\frac{\vdots}{1}$$

no boolean roots⟶ F is UNSAT

Monomial size: cumulative #monomials in the proof
Degree: largest degree among proof lines

Degree complexity of a PC derivation of p from F

$$\min\{d \text{ s.t. } p \text{ has a PC proof from } F \text{ of degree } d\}$$

Large degree → Large monomial size

Theorem (IPS, 1999)

Any k-CNF on n variables that requires degree d to be refuted in PC also requires PC refutations of monomial size at least

$$\exp\left(\frac{\Omega(d-k)^2}{n}\right)$$

Degree lower bound $\Omega(n)$ implies $2^{\Omega(n)}$ monomial size,

Small degree → small monomial size

Any formula of n variable refutable in degree d has a refutation

- of size n^d
- constructible in time $n^{O(d)}$

Proof.

E.g. Buchberger algorithms for Gröbner basis computation, limited to degree d. □

Small monomial size AND large degree

There are 3-CNFs on n variables that

- require degree $\Omega\left(\sqrt{n}\right)$ to be refuted [GL'10]
- have polynomial size refutation in PC

Cor 1. cannot improve the exponent of lower bound

$$\text{monomial size } \geqslant \exp\left(\frac{\Omega(d-k)^2}{n}\right)$$

Cor 2. The $n^{O(d)}$ proof search algorithm can be inefficient.

Degree lower bound d

Define **linear** operator $\mathcal{L}$ over polynomials in $\mathbb{F}$

- $\mathcal{L}(f_j) = 0$ and $\mathcal{L}(x_i^2 - x_i) = 0$
- if $\deg(p) < d$ then $\mathcal{L}(x_i p) = \mathcal{L}(x_i \mathcal{L}(p))$
- $\mathcal{L}(1) \neq 0$

$\mathcal{L}$ sets to 0 all polynomials derivable in degree $\leqslant d$, and 1 is not among them.

Degree lower bound d

Define **linear** operator $\mathcal{L}$ over polynomials in $\mathbb{F}$

- $\mathcal{L}(f_j) = 0$ and $\mathcal{L}(x_i^2 - x_i) = 0$
- if $\deg(p) < d$ then $\mathcal{L}(x_i p) = \mathcal{L}(x_i \mathcal{L}(p))$
- $\mathcal{L}(1) \neq 0$

$\mathcal{L}$ sets to 0 all polynomials derivable in degree $\leqslant d$, and 1 is not among them.

Some form of expansion in the formula allows to define such operator for large d. [AR'01][GL'10][MN'15]

v. memory issues

(quick mention)

Clause database

CDCL solvers learn a **massive** amount clauses.

Too many to be kept in memory:

- remove clauses to make space
- removed clauses may be useful to the proof
- which clauses to keep?

strategy to manage clause database

Blackboard model

A proof of f from $F = f_1 \wedge f_2 \wedge \ldots \wedge f_m$ of length t is

$$B_0 \quad B_1 \quad \ldots \quad B_{t-1} \quad B_t$$

where B_i is the content of the blackboard at time i.

- B_0 is emtpy
- B_t contains f.

Proof steps in the “blackboard model”

$$B_0 \quad B_1 \quad \ldots \quad B_{t-1} \quad B_t$$

At every step i either:

- **(axiom download)** $B_i = B_{i-1} \cup \{f_j\}$
- **(erasure)** $B_i \subseteq B_{i-1}$
- **(inference)** $B_i = B_{i-1} \cup \{g\}$ where $B_{i-1} \vdash g$.

If a formula is erased, if needed again must be re-derived.

Space measures: the “size” of the board

Resolution

- clauses, occurrences of literals

Polynomial calculus:

- monomials, polynomials

Cutting planes

- inequalities, cumulative coefficient bit-lengths

Questions about space complexity

Theoretical

- Space lower bounds
- Size/Space trade-offs
- Connection between width/degree and space

Practical

- How well does space measure memory in solvers?
- Is it a relevant measure of hardness?

vi. Extended resolution and DRAT

DRAT proofs

Resolution proofs capture basic CDCL solvers

- does not capture state-of-the-art pre/in-processing
- too verbose

DRAT proofs

Resolution proofs capture basic CDCL solvers

- does not capture state-of-the-art pre/in-processing
- too verbose

DRAT proofs [JHB'12][HHW'13]

- simulates resolution (hence CDCL generated proofs)
- simulates state-of-the-art pre/in-processing
- more compact
- includes description of erasures
- **stay tuned** for next talk.

Extended resolution (ER)

ER = Resolution + Extension axiom:

$$y_j \leftrightarrow \ell_1 \vee \ell_2 \vee \cdots \vee \ell_m$$

or equivalently

$$\neg y_j \vee \ell_1 \vee \ell_2 \vee \cdots \vee \ell_m \qquad \neg \ell_i \vee y_j \text{ for } i \in [m]$$

where ℓ_i are literals over

- initial variables
- extension variables $y_1, \ldots, y_{j-1}$.

Power of Extended resolution

Strength is connected to proof lines computational power

- clauses (resolution)
- linear inequalities (CP)
- polynomial equations (PC)
- bounded depth circuits (BD-Frege)
- formulas (Frege)
- boolean circuits (Extended Frege)

ER equivalent to Extended Frege, a **very strong** system

Unsatisfactory state of affair

DRAT and Extended resolution are **equivalent** [JHB'12][KRPH'18]

- no chances of proving unconditional lower bounds
- not many candidates for hard formulas

Impossible to say something relevant about modern SAT solvers using DRAT/ER as the reference proof system.

vii. proof search

Automatizability [Bonet, Pitassi, Raz, 1997]

Proof system P is **automatizable** when

- there is algorithm A : UNSAT $\rightarrow$ proofs in P
- $A(\phi)$ is a proof of ϕ in P;
- $A(\phi)$ runs in time $(|\phi| + |\pi|)^{O(1)}$;

where π is a smallest proof of ϕ in P.

A tentative approach

Width complexity $w \rightarrow$ resolution refutation in time $n^{O(w)}$

Degree complexity $d \rightarrow$ PC refutation in time $n^{O(d)}$

Are these proof search algorithms efficient?

A tentative approach

Width complexity $w \rightarrow$ resolution refutation in time $n^{O(w)}$

Degree complexity $d \rightarrow$ PC refutation in time $n^{O(d)}$

Are these proof search algorithms efficient?

These algorithms are tight for worst case [ALN'16]

Formulas with resolution/PC proofs of size $n^{O(1)}$ and $n^{\Omega(1)}$ width/degree complexity [BG'99][GL'10]

Non-Automatizability

Theorem (Alekhnovich, Razborov 2001)

Assuming $\mathrm{FPT} \neq \mathrm{W[P]}$, *neither resolution nor tree-like resolution are automatizable.*

- original proof has a stronger assumption
- fixed by [EGG '08]
- proved for Polynomial Calculus in [GL'10]

conclusions

Summary

1. Proof systems and lower bound techniques
   - resolution
   - cutting planes
   - polynomial calculus
2. Memory and space (quick mention)
3. Extended resolution and DRAT
4. Proof search
   - bounded width/degree proof search
   - non-automatizability

Read more…

On the Interplay Between Proof Complexity and SAT Solving [Nordström, ACM SIGLOG 2015]

A (Biased) Proof Complexity Survey for SAT Practitioners [Nordström, SAT 2014]

The end